Project 3 — SST-2 Sentiment Classification
README Report

Tiantian Zhao, Ao Chan, Longjie Zhang

May 10, 2025

1 Model Performance & Key Hyper-parameters

Table 1: Best dev/test scores and hyper-parameters

Model	**Core hyper-params**	**Dev Acc.**	**Test Acc.**	**Macro-F1**
Logistic Regression	C=4, L2, max_iter = 1000	0.823	0.821	0.82
Linear SVM	C=2, linear kernel	0.852	0.850	0.85
Random Forest	n_{estim}=500, depth = free	0.790	0.783	0.78
XGBoost	n_{estim}=400, depth = 6, η=0.1	0.865	0.862	0.86
FF-NN (GloVe frozen)	emb_dim = 100, hidden = 256, dropout = 0.5	0.847	0.844	0.84
FF-NN (GloVe trainable)	emb_dim = 100, hidden = 256, dropout = 0.5	0.861	0.859	0.86
BERT-base	lr = 2×10^{-5}, epochs = 3, batch = 16	0.931	0.927	0.93

2 Explanation of Key Choices (Detailed)

Table 2: Design decisions, gains and trade-offs

Stage	**Rationale**	**Δ Score**	**Trade-offs**
Vectoriser			
TF-IDF → GloVe → BERT	Bigrams capture short negations; GloVe densifies sparse text; contextual BERT resolves word sense.	+3 pp +2 pp +7 pp	Memory growth; BERT needs GPU.
Regularisers			
C (LR/SVM), weight_decay (BERT)	Swept $C \in [0.25, 8]$; 0.01 weight-decay curbed over-fit.	+1.8 pp +1.1 pp +0.6 pp	Larger search space; too much decay slows convergence.
Learning rate	Two-tier schedule for FF-NN; LR-finder chose 2×10^{-5} for BERT.	+1.5 pp +0.9 pp	Extra tuning epoch; tiny LR slows training.
Sequence length *max_length* = 60	Covers 95 % of SST-2 sentences; avoids truncating negations.	+0.4 pp	+9 % GPU memory.
Batch & GradAcc	Batch 16 + grad-acc 2 emulates batch 32 within 12 GB.	0	8 % longer wall-time.
Early stopping (patience = 2)	Prevents late-epoch drift.	0.6 pp variance	Risk of premature stop if patience too small.

3 Analysis

3.1 Parameter-tuning Workflow

1. **Prototype grid → random search**. A 3-parameter grid on 5 % data fixed sensible bounds; `RandomizedSearchCV` (30 trials) then found near-optimal XGBoost settings in under one hour.
2. **Nested CV**. 5-fold outer loops gave unbiased test estimates; 3-fold inner loops tuned C and tree depth.
3. **Progressive freezing**. Compared all-trainable vs. frozen vs. delayed-unfreeze (epoch 3). Delayed won by +0.3 pp but we chose all-trainable for reproducibility.
4. **LR-range test for BERT**. 200-step sweep revealed a flat minimum at 2×10^{-5}; used with 10 % linear warm-up and weight-decay 0.01.

3.2 Result Interpretation

- Classical TF-IDF + linear models plateau near 85 %—surface n-grams only.
- Tree ensembles add non-linear n-gram interactions (+1 pp) but risk memorisation.
- Static-embedding FF-NN exploits semantic similarity (-2 pp error); fine-tuning adds $+1\frac{1}{2}$ pp.
- BERT captures long-range context and negation; macro-F1 ↑ 0.93, dev-loss flattens after 3 epochs.

Table 3: Progression from classical ML to Transformers

Aspect	**Classical ML**	**Static-embed NN**	**Transformer**
Feature scope	Sparse n-grams	Dense global vectors	Contextual sub-words
Parameters	<1 M	$\sim$2 M	109 M
GPU time (V100)	— (CPU)	5 min	18 min
Accuracy	82–86 %	84–86 %	**92 %**
Interpretability	High	Medium	Low (needs attention maps)

4 Conclusion

Top performers. Fine-tuned **BERT-base** (92.7 % / 0.93) and **XGBoost** (86.2 % / 0.86) lead their categories.

Why BERT wins: contextual embeddings resolve clause-internal negation and leverage 330 M-token pre-training.

Where XGBoost shines: $<$ 1 ms CPU inference and ranked feature importance support edge deployment and error analysis.

Sentiment insights. BERT's mean positive-class probability is 0.64 ($\pm$0.21), aligning with dataset balance; errors gather on sarcasm and idioms.

Take-aways:

1. Deep representation learning yields $\approx$7 pp gain, but demands LR scheduling and memory-aware batching.
2. Classical models remain viable when compute or transparency is paramount.
3. BERT probabilities can serve directly as a high-resolution sentiment index in down-stream dashboards.